

Claims:

1       1. A system for executing application software on a operating system within a  
2 secured run-time environment without affecting an application software resources of a client  
3 computer, the system comprising:

4           an application wrapper, wherein said application wrapper shields the application software  
5 resources, whereby said secured run-time environment for executing said application  
6 software is created and the application software resources are protected; and

7           the application wrapper further comprising a privatized virtual file resource created from  
8 an operating system file system, a privatized virtual registry created from an operating  
9 system registry system, a privatized operating system shared component resource, a  
10 privatized application configuration resource and a privatized environmental resources  
11 for application variables.

12       2. The system of claim 1, wherein privatized virtual file resource further comprising:

13           intercepting file I/O request generated by one or more processes;

14           establishing a process ID for the intercepted file I/O request;

15           comparing process ID to establish operating system process and secured run-time  
16 process;

17           establishing a process ID as operating system process and secured run-time process;

18           servicing the file I/O request for all process ID established as secured run-time process;

19           redirecting the file I/O request to operating system service for process ID established as  
20 operating system process;

21           rejecting the file I/O request on secured run-time process resources for process ID  
22 established as operating system process;

23           comparing process ID established as secured run-time process to further establish process  
24 resources corresponding to process ID;

25           establishing corresponding process resources within secured run-time resources; and

26       rejecting the file I/O request on secured run-time process resources for process ID  
27       established as secured run-time process and process ID belongs to other process  
28       resources.

29       3.       The system of claim 1, wherein privatized virtual registry further comprises:  
30           privatizing virtual registry system by intercepting registry I/O request generated by  
31           several process;

32           establishing process ID for the intercepted registry I/O request;  
33           comparing process ID to establish operating system process and secured run-time  
34           process;

35           establishing process ID as operating system process and secured run-time process;  
36           servicing the registry I/O request for all process ID established as secured run-time  
37           process;

38           redirecting the registry I/O request to operating system service for process ID established  
39           as operating system process;

40           rejecting the registry I/O request on secured run-time process resources for process ID  
41           established as operating system process;

42           comparing process ID established as secured run-time process to further establish process  
43           resources corresponding to process ID;

44           establishing corresponding process resources within secured run-time resources; and

45           rejecting the registry I/O request on secured run-time process resources for process ID  
46           established as secured run-time process and process ID belongs to other process resources.

47       4.       The system of claim 1, wherein privatizing operating system shared component  
48       resource further comprising:

49           searching secured application process for injecting component hooker;  
50           checking the said secured application process to establish whether the process is injected  
51           with component hooker;

52 establishing the said secured application process as new secured application process for  
53 said secured application process not injected with component hooker;

54 injecting component hooker to new secured application process to intercept component  
55 process;

56 repeating component hooker injection for all the new secured application process;

57 5. The system of claim 1, wherein privatizing operating system shared component  
58 resource further comprising:

59 Initializing component redirection table to provide component redirecting information;

60 Registering virtual component required for the secured application;

61 Adding redirecting information to the said component redirection table for the execution  
62 of each selected said secured run-time application;

63 Removing component redirecting information from the said component redirection table  
64 for the termination of each said secured run-time application;

65 6. The system of claim 1, wherein privatizing operating system shared component  
66 resource further comprising:

67 intercepting component process function for replacing component search path with  
68 secured application resource path;

69 replacing component search path with private resource path to load the component from  
70 the secured application resource path; and

71 creating new process for the intercepted component with the replaced secured application  
72 resource path.

73 7. The system of claim 1, wherein privatizing operating system shared component  
74 resource further comprising:

75 intercepting component call for replacing component registry path with the said  
76 privatized virtual registry path;

77 searching component redirection table for the said component redirecting information;

78       replacing component registry path with the said privatized virtual registry path retrieved  
79       from the said component redirection table;

80       returning the intercepted call to the requested call with the replaced secured application  
81       registry path for addressing the component location from the privatized virtual registry  
82       system and further the said component is addressed to load from the said privatized  
83       virtual file system;

84       redirecting the said component call as it is to the requested call for the said component  
85       call originated from non secured run-time application and for the said component call,  
86       which do not have redirecting information in the said component redirection table.

87       8.       The system of claim 1, wherein privatizing operating system shared component  
88       resource further comprising:

89       intercepting the said RPC message call for replacing component information with  
90       privatized virtual component information;

91       searching component redirecting information from the said component redirection table;

92       replacing RPC message with the said privatized virtual component information retrieved  
93       from the said component redirection table;

94       returning the intercepted RPC message call to the requested call with the replaced  
95       message;

96       continuing the RPC call to locate the privatized virtual component through SCM;

97       redirecting the said RPC message call as it is to the requested call for the said component  
98       call originated from non secured run-time application and for the said component call,  
99       which do not have redirecting information in the said component redirection table.

100       9.       The system of claim 1, wherein privatized application configuration resource  
101       further comprises:

102       monitoring file I/O request for configuration file to provide separate configuration file;

103       searching and retrieving configuration file from secured application resources; and

104       serving application configuration file to requesting process.

105           10. The system of claim 1, wherein privatized environmental resources further  
106 comprises:

107           intercepting environment variable request to supply private values to secured application  
108           process;

109           verifying process ID to establish the process ID as operating system process or secured  
110           application process;

111           redirecting the call for process ID established as operating system process;

112           reading variable data from secured application resource and returning the value to  
113           requested process for read variable calls requested by the secured application process;

114           searching the requesting write variable in secured application resource to find the  
115           presence of requesting write variable;

116           creating variable with variable data within secured application resource and returning the  
117           status to requested process for variable do not exist in secured application resource; and  
118           updating variable with variable data within secured application resource and returning the  
119           status to requested process for variable exist in secured application resource.

120           11. The system of claim 1, wherein the application wrapper further comprises  
121           selectively allowing the application software to interact operating system resources directly  
122           during the said application software executing under the said secured run-time environment.

123           12. The system of claim 1, wherein the application wrapper further comprises  
124           selectively allowing said application software to interact with other application software  
125           resources directly during the said application software executing under the said secured run-time  
126           environment.

127           13. The system of claim 1, wherein said application wrapper further comprises  
128           providing a run-time environment to said application software that is visible to an operating  
129           system run-time environment without having said application software run-time resources,  
130           whereby said application software resources is simulated to said secured run-time environment  
131           during the execution of said application software.

132        14. The system of claim 13, further comprising means for protecting the behavior of  
133        said application software from other application and said operating system.

134        15. The system of claim 13, further comprising means for eliminating said application  
135        conflicts from other running application software.

136        16. The system of claim 13, further comprising means for executing multiple instance  
137        of single said application software.

138        17. The system of claim 1, wherein the said application wrapper further comprising  
139        maintaining the application software resources away from said operating system resources,  
140        whereby said operating system resources is protected from said application software resources.

141        18. The system of claim 1, wherein said application wrapper further comprises  
142        permitting full access to said application software that requires to access for variation occurs to  
143        said application software resources within the said application wrapper.

144        19. The system of claim 18 further comprising means for keeping the state of secured  
145        run-time environment to said application software.

146        20. The system of claim 18, further comprising means for updating said application  
147        software resources required by said application software.

148        21. The system of claim 1, wherein the said application wrapper monitors the said  
149        application run-time request to determine the required said application software resources for  
150        execution.

151        22. The system of claim 21, further comprising means for receiving said application  
152        software resources to execute said application software in the said secured run-time environment.

153        23. The system of claim 21, further comprising means for incrementally executing the  
154        said application software in the secured run-time environment.

155        24. A method for executing application software on a operating system within a  
156        secured run-time environment without affecting an application software resources of a client  
157        computer, the client compute comprising an application wrapper, wherein said application  
158        wrapper shields the said application software resources, whereby a said secured run-time

159 environment for executing an said application software is created and the said application  
160 software resources is protected, the method further comprising:

161 privatizing virtual file resource created from an operating system file system;

162 privatizing virtual registry created from an operating system registry system;

163 privatizing operating system shared component resource;

164 privatizing application configuration resource; and

165 privatizing environmental resources for application variables.

166 25. The method of claim 24, wherein privatizing the virtual file resource further  
167 comprising:

168 intercepting file I/O request generated by several processes;

169 establishing process ID for the intercepted file I/O request;

170 comparing process ID to establish operating system process and secured run-time  
171 process;

172 establishing process ID as operating system process and secured run-time process;

173 servicing the file I/O request for all process ID established as secured run-time process;

174 redirecting the file I/O request to operating system service for process ID established as  
175 operating system process;

176 rejecting the file I/O request on secured run-time process resources for process ID  
177 established as operating system process;

178 comparing process ID established as secured run-time process to further establish process  
179 resources corresponding to process ID;

180 establishing corresponding process resources within secured run-time resources; and

181 rejecting the file I/O request on secured run-time process resources for process ID  
182 established as secured run-time process and process ID belongs to other process  
183 resources.

184        26. The method of claim 24, wherein Privatizing the virtual registry further  
185 comprising:

186        privatizing virtual registry system by intercepting registry I/O request generated by  
187        several process;

188        establishing process ID for the intercepted registry I/O request;

189        comparing process ID to establish operating system process and secured run-time  
190        process;

191        establishing process ID as operating system process and secured run-time process;

192        servicing the registry I/O request for all process ID established as secured run-time  
193        process;

194        redirecting the registry I/O request to operating system service for process ID established  
195        as operating system process;

196        rejecting the registry I/O request on secured run-time process resources for process ID  
197        established as operating system process;

198        comparing process ID established as secured run-time process to further establish process  
199        resources corresponding to process ID;

200        establishing corresponding process resources within secured run-time resources; and

201        rejecting the registry I/O request on secured run-time process resources for process ID  
202        established as secured run-time process and process ID belongs to other process  
203        resources.

204        27. The method of claim 24, wherein privatizing operating system shared component  
205        resource further comprising:

206        intercepting component process function for replacing component search path with  
207        secured application resource path;

208        replacing component search path with private resource path to load the component from  
209        the secured application resource path; and

210       creating new process for the intercepted component with the replaced secured application  
211       resource path.

212       28.    The method of claim 24, wherein privatizing operating system shared component  
213       resource further comprising:

214       searching secured application process for injecting component hooker;  
215       checking the said secured application process to establish whether the process is injected  
216       with component hooker;  
217       establishing the said secured application process as new secured application process for  
218       said secured application process not injected with component hooker;  
219       injecting component hooker to new secured application process to intercept component  
220       process;  
221       repeating component hooker injection for all the new secured application process;

222       29.    The method of claim 24, wherein privatizing operating system shared component  
223       resource further comprising:

224       Initializing component redirection table to provide component redirecting information;  
225       Registering virtual component required for the secured application;  
226       Adding redirecting information to the said component redirection table for the execution  
227       of each selected said secured run-time application;  
228       Removing component redirecting information from the said component redirection table  
229       for the termination of each said secured run-time application;

230       30.    The method of claim 24, wherein privatizing operating system shared component  
231       resource further comprising:

232       intercepting component call for replacing component registry path with the said  
233       privatized virtual registry path;  
234       searching component redirection table for the said component redirecting information;  
235       replacing component registry path with the said privatized virtual registry path retrieved  
236       from the said component redirection table;

237        returning the intercepted call to the requested call with the replaced secured application  
238        registry path for addressing the component location from the privatized virtual registry  
239        system and further the said component is addressed to load from the said privatized  
240        virtual file system;

241        redirecting the said component call as it is to the requested call for the said component  
242        call originated from non secured run-time application and for the said component call,  
243        which do not have redirecting information in the said component redirection table.

244        31.      The method of claim 24, wherein privatizing operating system shared component  
245        resource further comprising:

246        intercepting the said RPC message call for replacing component information with  
247        privatized virtual component information;

248        searching component redirecting information from the said component redirection table;  
249        replacing RPC message with the said privatized virtual component information retrieved  
250        from the said component redirection table;

251        returning the intercepted RPC message call to the requested call with the replaced  
252        message;

253        continuing the RPC call to locate the privatized virtual component through SCM;

254        redirecting the said RPC message call as it is to the requested call for the said component  
255        call originated from non secured run-time application and for the said component call,  
256        which do not have redirecting information in the said component redirection table.

257        32.      The method of claim 24, wherein privatizing application configuration resource  
258        further comprising:

259        monitoring file I/O request for configuration file to provide separate configuration file;

260        searching and retrieving configuration file from secured application resources; and

261        serving application configuration file to requesting process.

262        33.      The method of claim 24, wherein privatizing environmental resources for  
263        application variables further comprising:

264 intercepting environment variable request to supply private values to secured application  
265 process;  
266 verifying process ID to establish the process ID as operating system process or secured  
267 application process;  
268 redirecting the call for process ID established as operating system process;  
269 reading variable data from secured application resource and returning the value to  
270 requested process for read variable calls requested by the secured application process;  
271 searching the requesting write variable in secured application resource to find the  
272 presence of requesting write variable;  
273 creating variable with variable data within secured application resource and returning the  
274 status to requested process for variable do not exist in secured application resource; and  
275 updating variable with variable data within secured application resource and returning the  
276 status to requested process for variable exist in secured application resource.

277       34. The method of claim 24, wherein selectively allows said application software to  
278 interact operating system resources directly during the said application software executing under  
279 the said secured run-time environment.

280        35. The method of claim 24, wherein selectively allows said application software to  
281 interact with other application software resources directly during the said application software  
282 executing under the said secured run-time environment.

283       36. The method of claim 24, wherein said application wrapper provides an run-time  
284 environment to said application software that visible to be an operating system run-time  
285 environment without having said application software run-time resources, whereby said  
286 application software resources is simulated to said secured run-time environment during the  
287 execution of said application software.

37. The method of claim 36, further comprising protecting the behavior of said application software from other application and said operating system.

290       38. The method of claim 36, further comprising eliminating said application conflicts  
291 from other running application software.

292       39. The method of claim 36, further comprising executing multiple instance of single  
293 said application software.

294       40. The method of claim 24, wherein the said application wrapper keeps the  
295 application software resources away from said operating system resources, whereby said  
296 operating system resources is protected from said application software resources.

297       41. The method of claim 24, wherein said application wrapper allows full access to  
298 said application software that requires to access for variation occurs to said application software  
299 resources within the said application wrapper.

300       42. The method of claim 41, further comprising a means for keeping the state of  
301 secured run-time environment to said application software.

302       43. The method of claim 41, further comprising a means for updating said application  
303 software resources required by said application software.

304       44. The method of claim 24, wherein the said application wrapper monitors the said  
305 application run-time request to determine the required said application software resources for  
306 execution.

307       45. The method of claim 44, further comprising a means for receiving said  
308 application software resources to execute said application software in the said secured run-time  
309 environment.

310       46. The method of claim 44, further comprising a means for incrementally executing  
311 the said application software in the secured run-time environment.